

Beamformed observing modes

Cees Bassa

ASTRON

March 23, 2021

Outline

- 1 Beamforming in a nutshell
- 2 LOFAR beamforming highlights
- 3 Beams and LOFAR
- 4 COBALT correlator and beamformer
- 5 Observation configurations
- 6 COBALT output
- 7 Some caveats
- 8 Tutorial

Beamforming in a nutshell

Q: What is beamforming?

A: Adding signals from different antennas in phase

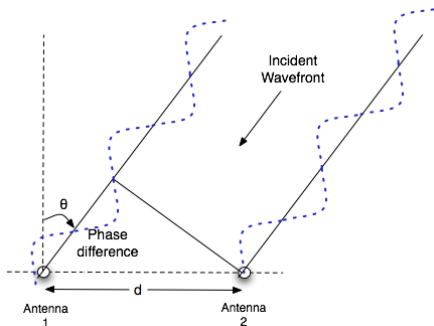
(Note that correlation is a multiplication)

Q: Why beamform?

A: Increase sensitivity of your telescope

Also referred to as:

- coherent sum
- coherent addition
- phased array
- tied array



source: wikipedia

Using LOFAR as a single dish telescope

Interferometry

- correlates antenna signals
- high spatial resolution
- low time resolution

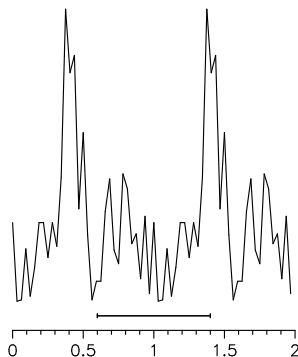
Beamforming

- adds antenna signals
- low spatial resolution
- high time resolution

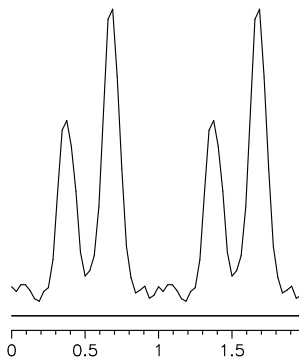
- Beamforming trades spatial resolution for time resolution
- Much easier than interferometry; e.g. no phase/amplitude calibration, deconvolution etc. . .

Recent LOFAR beamforming results

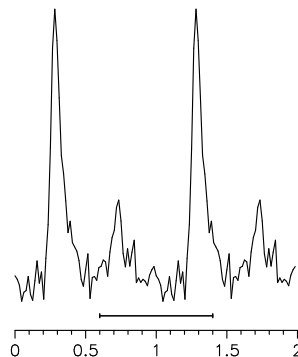
J0653+4706



J0952-0607



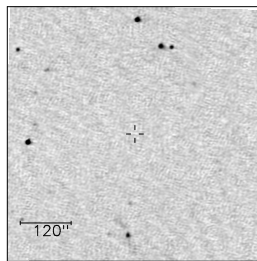
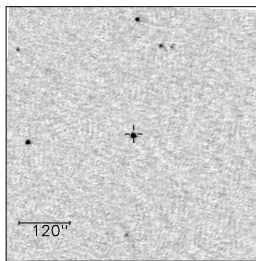
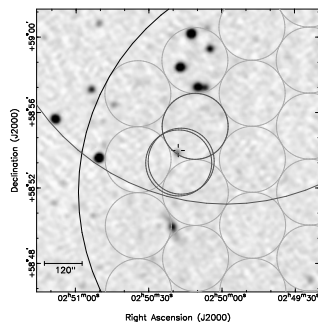
J1552+5437



Pulse phase

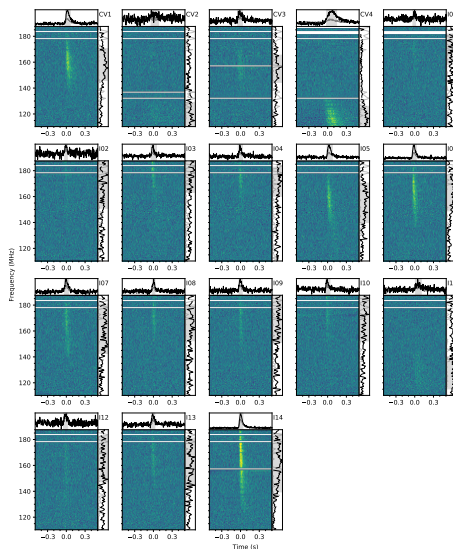
source: Bassa et al. (2017)

Recent LOFAR beamforming results



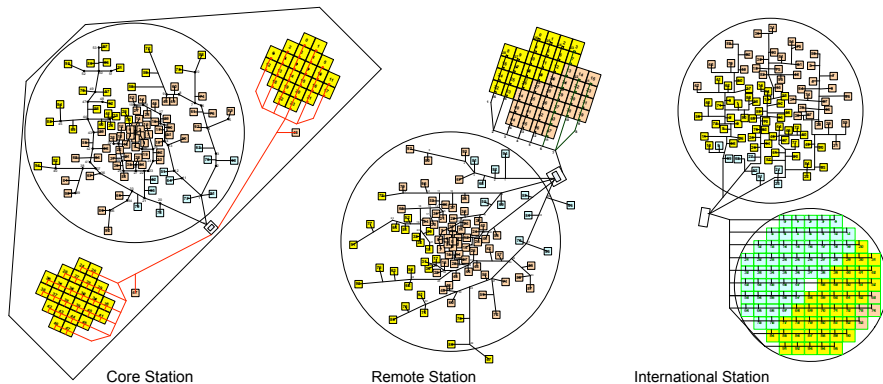
source: Tan et al. (2018)

Recent LOFAR beamforming results



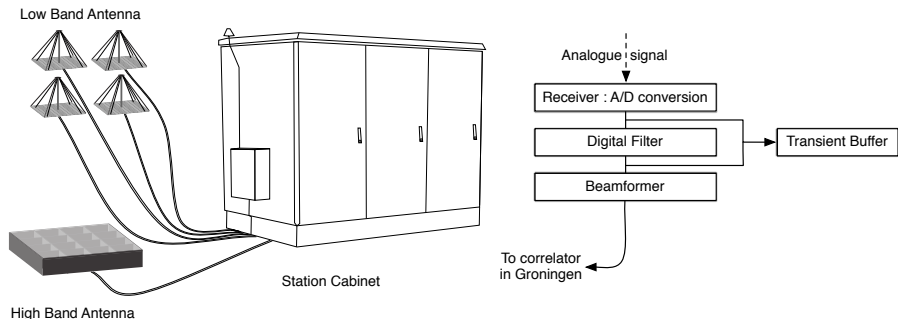
source: Pleunis et al. (2021)

LOFAR stations (a review)



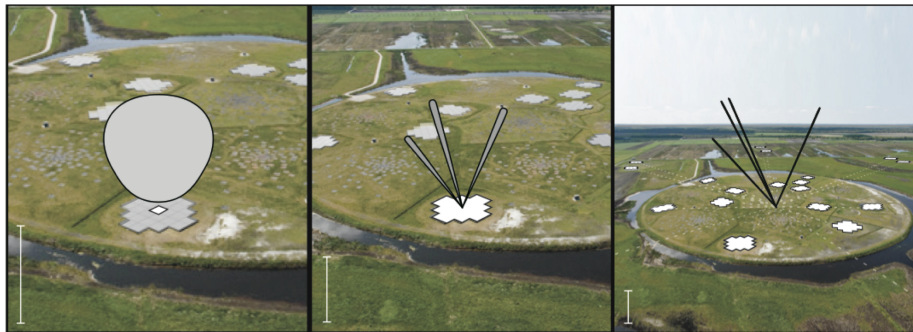
van Haarlem et al. 2013

LOFAR stations (a review)



van Haarlem et al. 2013

Beam terminology



source: astron

Element beam
or
Tile beam

Station beam
or
Sub-array pointing (SAP)

Array beam
or
Tied-array beam (TAB)

Tile beams

Tile beamforming:

- Analog delay lines
- 5 bit delays of 0.5 ns
- HBA tiles only
- *Summator* combines 4×4 HBA dipoles
- $\lambda/D \sim 25^\circ$ for $\lambda = 2.2$ m, $D = 5$ m
- Updated once every few minutes



source: max planck

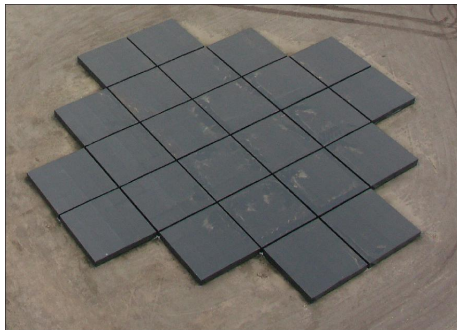
$$\Delta t = (1b_1 + 2b_2 + 4b_3 + 8b_4 + 16b_5) \times 0.5 \text{ ns}$$

e.g. 11111 \rightarrow 15.5 ns = 4.65 m, or 10110 \rightarrow 6.5 ns = 1.95 m

Station beams (aka sub-array pointings)

Station beamforming:

- Digital signals from 48 HBA tiles or 48 LBA dipoles
- Polyphase filter over 160 or 200 MHz to 512 subbands
- 16 or 8 bit digital representation
- 244 or 488 beam/subband combinations (aka beamlets)
- Phase-rotation beamformer
- Updates every second



source: astron

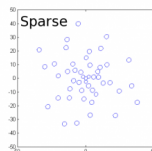
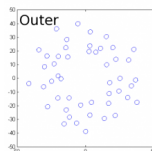
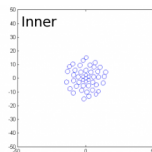
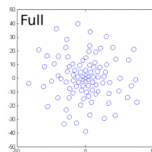
Stations configurations

LBA:

- **OUTER:** Outer 48 antennas
- **INNER:** Inner 48 antennas
- **SPARSE:** Sparse config
- **X or Y:** Single polarization from 96 antennas

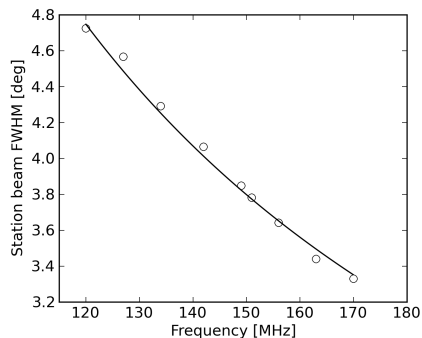
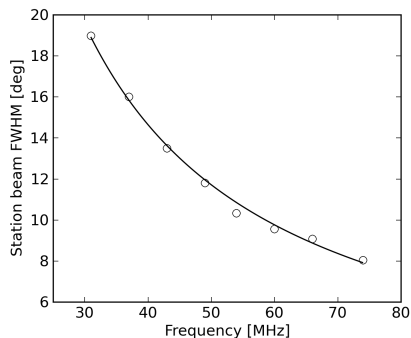
HBA:

- **DUAL:** Substations separately
- **JOINED:** Substations together
- **0 or 1:** Single substation



source: astron

Station beam sizes



source: van Haarlem et al. (2013)

LBA_INNER and a single **HBA** core substation

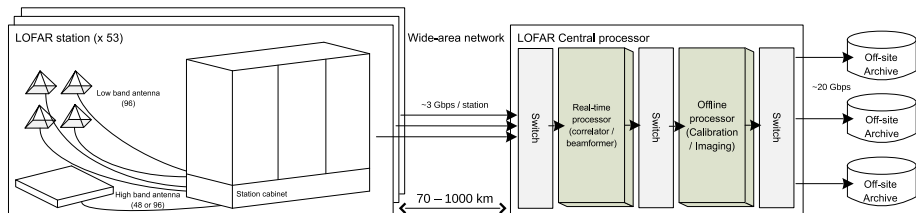
COBALT

COBALT: COrrellator and Beamformer Application for the LOFAR Telescope

- Using CPUs and GPUs
- Replaced Blue Gene in 2014
- Located in Groningen
- Hardware upgrade in 2019 (**COBALT2.0**)
- 11 nodes + 2 spare/testing
- Software upgrade in 2021



COBALT signal path

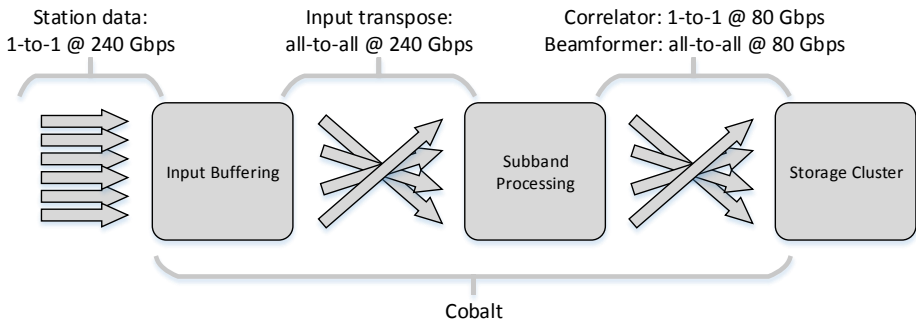


source: Broekema et al. (2018)

- 200 MHz clock, 195.3125 kHz channels, $5.12 \mu\text{s}$ samples
- 160 MHz clock, 156.250 kHz channels, $6.4 \mu\text{s}$ samples

Data rate: $244 \text{ beamlets} \times 16 \text{ bits} \times 2 \text{ polarizations} \times 2 \text{ values per sample} \times 195312.5 \text{ samples s}^{-1} = 3.05 \text{ Gb s}^{-1}$.

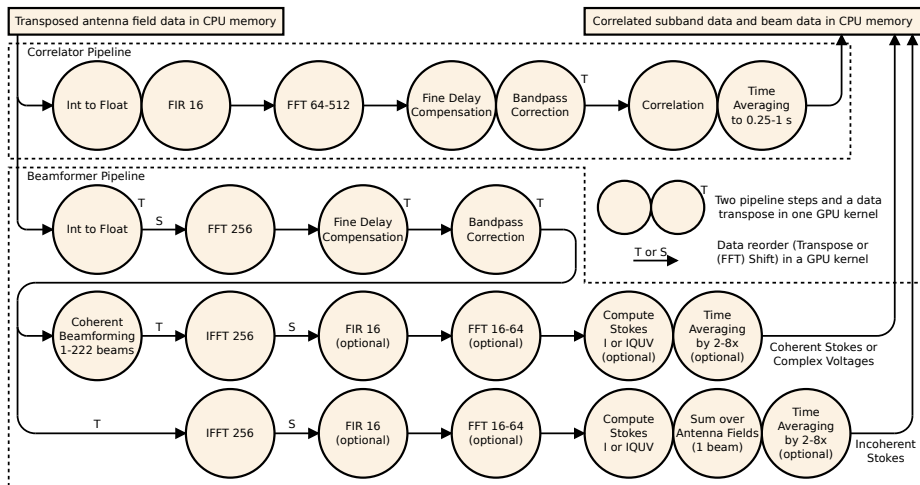
COBALT data flow



source: Broekema et al. (2018)

- **COBALT** designed to handle 80 stations at 3 Gb s^{-1} , 240 Gb s^{-1} total.

COBALT processing



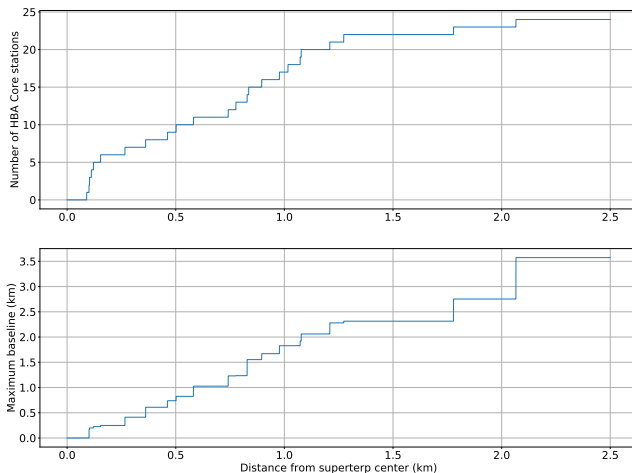
source: Broekema et al. (2018)

COBALT Configuration

Many configurations to choose from:

- Choice of stations
- Coherent sum or incoherent sum
- Fly's eye (FE; each station independent)
- Observing mode: full Stokes (IQUV), Stokes I (I) or complex voltage (XXYY)
- Number of beams (tied-array rings)
- Sub-band selection
- Channels per subband (1, 16, 32, 64, 128, 512)
- Downsampling factor

Which stations? Sensitivity vs beamsize



$$S \propto N_{\text{station}} \text{ and } \theta \sim \lambda/D_{\text{max}}$$

Coherent vs incoherent

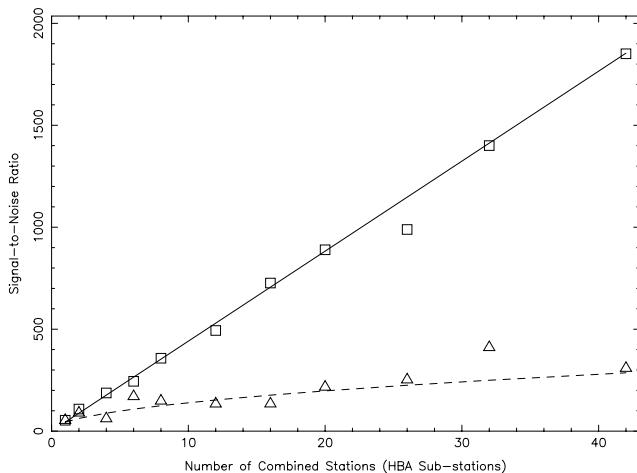
Coherent addition

- Sum *voltages* (V)
- Phase information retained
- $\text{SNR} \propto N_{\text{station}}$
- Tied-array beamsizes
- Complex voltage (XXYY) or coherent Stokes (CS) output

Incoherent addition

- Sum *powers* P ($P \propto V^2$)
- Phase information lost
- $\text{SNR} \propto \sqrt{N_{\text{station}}}$
- Station beamsizes
- Incoherent Stokes (IS) output

Coherent vs incoherent



source: van Haarlem et al. (2013)

Complex voltages or Stokes parameters

Complex voltages (XXYY)

- Complex number for each polarization: $\vec{e} = e_x + ie_y$
- Two polarizations, so four values per sample
- Sampled at the Nyquist rate (5.12 μ s for 200 MHz clock, 6.4 μ s for 160 MHz clock)

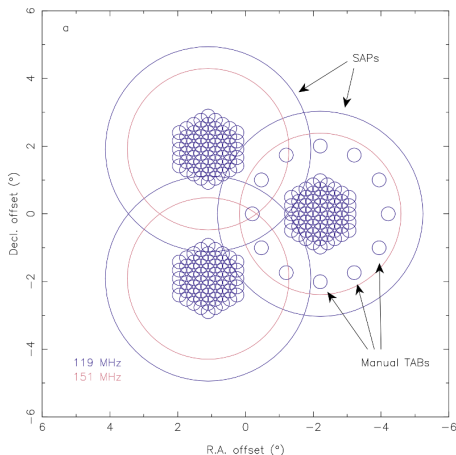
Stokes parameters (CS or IS)

- $I = \langle |e_x|^2 \rangle + \langle |e_y|^2 \rangle$
- $Q = \langle |e_x|^2 \rangle - \langle |e_y|^2 \rangle$
- $U = 2\text{Re} \left| \langle e_y e_x^* \rangle \right|$
- $V = 2\text{Im} \left| \langle e_y e_x^* \rangle \right|$
- Time averaging possible
- Can select $IQUV$ or just I

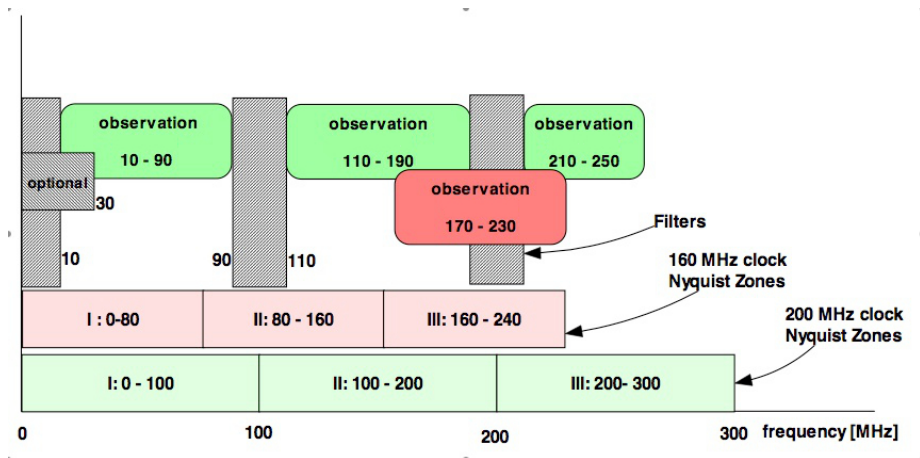
Number of beams?

Configuration options:

- Manual placement *or*
- hexagonal tied-array rings
- rings in α , δ coordinates
- 1, 7, 19, 37, 61, 91...
- Can be defined per sub-array pointing



Sub-band selection



source: astron

Sub-band selection

Configuration options:

- Sampler clock;
 $\nu_{\text{clk}} = 200 \text{ MHz or } 160 \text{ MHz}$
- Nyquist zone; $n = 1, 2 \text{ or } 3$
- Subband numbers;
 $s = 0 \dots 244$ for 16 bit, or
 $s = 0 \dots 488$ for 8 bit
- subband \rightarrow frequency:
$$\nu = \left(n - 1 + \frac{s}{512}\right) \frac{\nu_{\text{clk}}}{2}$$
- frequency \rightarrow subband:
$$s = \left\lfloor \frac{1024}{\nu_{\text{clk}}} \left(\nu - \frac{(n-1)\nu_{\text{clk}}}{2}\right) \right\rfloor$$

Estimating data rates

$$r = n_{\text{beam}} \times n_{\text{sub}} \times n_{\text{chan}} \times n_{\text{value}} \times n_{\text{bit}} / (n_{\text{chan}} \times n_{\text{downsamp}} \times t_{\text{samp}})$$

- n_{beam} : beams
- n_{sub} : subbands
- n_{chan} : channels per subband
- n_{value} : values per sample
- n_{bit} : bits (32 by default)
- n_{downsamp} : downsampling factor
- t_{samp} : sampling time (5.12 μs or 6.4 μs)

	t_{samp} (μs)	n_{beam}	n_{sub}	n_{chan}	n_{value}	n_{bit} (bit)	n_{downsamp}	r (Gbit s $^{-1}$)
LOTAAS	5.12	222	162	16	1	32	6	37.5
MSP	5.12	7	200	1	4	32	1	35.0
Timing	5.12	1	400	1	4	32	1	10.0

COBALT limits

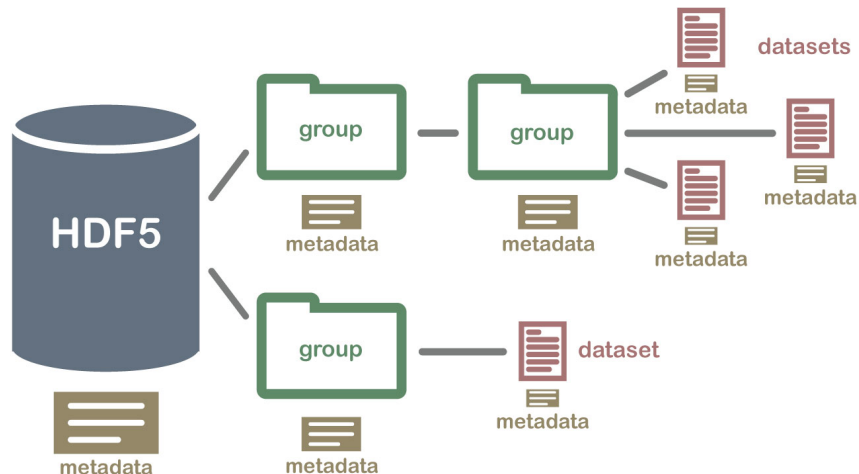
Limits:

- $r < 80 \text{ Gbit s}^{-1}$
- Higher throughput if processing is parallel
- Combination of n_{station} , n_{sub} , n_{beam} , n_{chan} , n_{downsamp} can be fine tuned
- Contact science support for questions
- Offline and online tests can be performed
- Improved capabilities with COBALT 2.0 software next year

```
AppCtrl.application=CorrAppl
AppCtrl.processes=[CorrProc]
AppCtrl.resultfile=/opt/lofar/var/run/ACC_CCU001:OnlineControl[0][666002]_CorrAppl_result.param
Cobalt.BeamFormer.CoherentStokes.nrChannelsPerSubband=16
Cobalt.BeamFormer.CoherentStokes.subbandsPerFile=512
Cobalt.BeamFormer.CoherentStokes.timeIntegrationFactor=6
Cobalt.BeamFormer.CoherentStokes.which=1
Cobalt.BeamFormer.IncoherentStokes.nrChannelsPerSubband=16
Cobalt.BeamFormer.IncoherentStokes.subbandsPerFile=512
Cobalt.BeamFormer.IncoherentStokes.timeIntegrationFactor=6
Cobalt.BeamFormer.IncoherentStokes.which=1
Cobalt.BeamFormer.coherentDisperseChannels=false
Cobalt.BeamFormer.flyEye=false
Cobalt.BeamFormer.nrDelayCompensationChannels=256
Cobalt.BeamFormer.nrHighResolutionChannels=256
Cobalt.BeamFormer.stationList=[]
Cobalt.Correlator.integrationTime=1.00663
Cobalt.Correlator.nrBlocksPerIntegration=1
Cobalt.Correlator.nrChannelsPerSubband=16
Cobalt.Correlator.nrIntegrationsPerBlock=1
Cobalt.FinalMetaDataGatherer.database.host=sasdb.control.lofar
Cobalt.FinalMetaDataGatherer.database.name=
Cobalt.FinalMetaDataGatherer.database.port=
Cobalt.FinalMetaDataGatherer.database.username=
Cobalt.FinalMetaDataGatherer.enabled=true
Cobalt.Nodes=[ cbt001_0, cbt001_1, cbt002_0, cbt002_1, cbt003_0, cbt003_1, cbt004_0, cbt004_1, cbt005_0,
cbt005_1, cbt006_0, cbt006_1, cbt007_0, cbt007_1, cbt008_0, cbt008_1 ]
Cobalt.OutputProc.StaticMetaDirectory=/data/home/lofarsys/production/lofar_cobalt/etc
Cobalt.OutputProc.executable=/outputProc
Cobalt.OutputProc.sshPrivateKey=
Cobalt.OutputProc.sshPublicKey=
Cobalt.OutputProc.userName=
Cobalt.PVSSGateway.host=ccu001
Cobalt.blockSize=196608
Cobalt.commandStreamFile=/localhome/lofar/lofar_versions/LOFAR-Release-3_2_0/var/run/rtpc-666002.pipe
Cobalt.correctBandPass=true
Cobalt.correctClocks=true
Cobalt.delayCompensation=true
Cobalt.realTime=true
CorrAppl.CorrProc.executable=CN_Processing
CorrAppl.CorrProc.hostname=cbmaster
CorrAppl.CorrProc.nodes=[]
CorrAppl.CorrProc.startstopType=bgl
CorrAppl.CorrProc.workingdir=/opt/lofar/bin/
CorrAppl.hostname=cbmaster
CorrAppl.extraInfo=["PIC", "Cobalt"]
CorrAppl.processOrder=[]
CorrAppl.processes=["CorrProc"]
DRAGNET.Nodes=[ drg01, drg02, drg03, drg04, drg05, drg06, drg07, drg08, drg09, drg10, drg11, drg12, drg13,
drg14, drg15, drg16, drg17, drg18, drg19, drg20 ]
Observation.AnaBeam[0].angle1=6.996355752001689
Observation.AnaBeam[0].angle2=0.0
```


Beamformed COBALT output

HDF5: Hierarchical Data Format (version 5)



source: hdf group

COBALT BF filename convention

L[nnnnnnn]_SAP[sss]_B[bbb]_S[s]_P[ppp]_bf.{h5, raw}

- **h5**: HDF5 header (~ 1 MB); contains header information
- **raw**: Raw data (many GBs); contains raw data
- **[nnnnnnn]**: Observation ID (ObsID or SASID)
- **[sss]**: Sub-array pointing number (SAP)
- **[bbb]**: Tied-array beam number
- **[s]**: Stokes IQUV parameter or complex voltage identifier (real X, imag X, real Y, imag Y)
- **[ppp]**: Frequency part (multiple subbands in one file)

Example: L650501_SAP000_B002_S2_P010_bf.h5

Reading HDF5

Options:

- h5dump, h5ls on linux command line to read header
- h5py python reader (will use in tutorials)
- pytables python reader
- LOFAR-DAL (Data Access Library) C++ library written by ASTRON
<https://github.com/nextgen-astrodata/DAL>
- dspsr pulsar software (lecture by Vlad Kondratiev)
- Plain old fopen on raw files (32 bit float or 8 bit char)

HDF5 for Python

[Downloads](#) [Documentation](#) [GitHub Project](#)



About the project

The h5py package is a Pythonic interface to the [HDF5](#) binary data format.

It lets you store huge amounts of numerical data, and easily manipulate that data from NumPy. For example, you can slice into multi-terabyte datasets stored on disk, as if they were real NumPy arrays. Thousands of datasets can be stored in a single file, categorized and tagged however you want.

H5py uses straightforward NumPy and Python metaphors, like dictionary and NumPy array syntax. For example, you can iterate over datasets in a file, or check out the `.shape` or `.dtype` attributes of datasets. You don't need to know anything special about HDF5 [to get started](#).

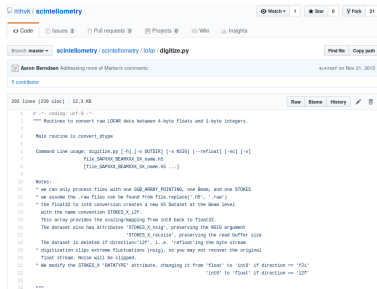
In addition to the easy-to-use high level interface, h5py rests on an object-oriented Cython wrapping of the HDF5 C API. Almost anything you can do from C in HDF5, you can do from h5py.

Best of all, the files you create are in a widely-used standard binary format, which you can exchange with [other people](#), including those who use programs like IDL and MATLAB.

Redigitizing HDF5 complex voltages

digitize.py by Marten van Kerkwijk

- Convert 32 bit float to 8 bit integers (256 levels)
- Only to be used on complex voltages (XXYY)
- 4× decrease in file size
- Stores scales and offsets in new HDF5 files
- Option offered by RO processing as part of PuLP (lecture by Vlad Kondratiev)



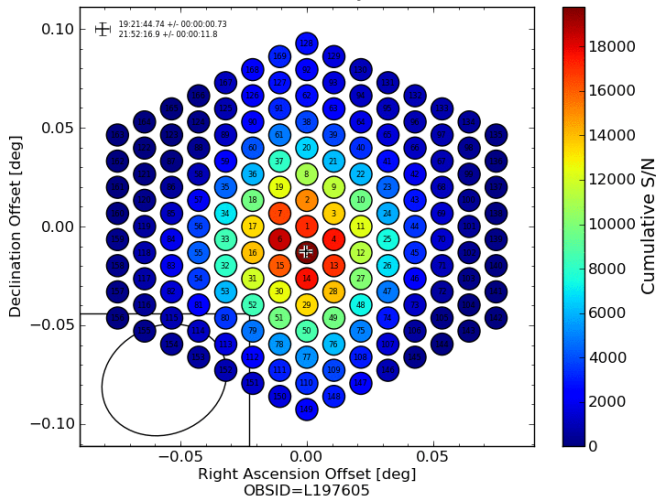
The screenshot shows the GitHub repository page for `mhvk/scintellometry`. The file `digitize.py` is selected, showing its code. The code is a Python script that converts raw LOFAR data from 4-byte floats to 8-bit integers. It includes a docstring, a main routine, and a command line usage example. The code is licensed under the MIT license.

```
1 # -*- coding: utf-8 -*-
2 """ Routines to convert raw LOFAR data between 4-byte floats and 8-bit integers.
3
4 Main routine is convert_dtype
5
6 Command line usage: digitize.py [-h] [-o OUTDIR] [-s NCHAN] [--refloat] [-nc] [-v]
7 file_SAP000_000000_00_name.h5
8 [file_SAP000_000000_00_name.h5 ...]
9
10 Notes:
11 * We can only process files with one SVD, 4096V, POINTING, one Beam, and one STORMS
12 * We assume the /raw files can be found from file.replace('.h5', '.raw')
13 * The float32 to int8 conversion creates a new h5 dataset at the Beam level
14 with the name conversion STORMS_X_127.
15 This array provides the scaling/mapping from jset back to F3MAG32.
16 The dataset also has attributes 'STORMS_X_min', preserving the N50 argument
17 'STORMS_X_max', preserving the read buffer size
18 'STORMS_X_min', i.e., refloating the byte stream
19 digitization clips extreme fluctuations (n50), so you may not recover the original
20 float stream. Noise will be clipped.
21 * We modify raw STORMS_X_127 attribute, changing it from 'float' to 'int8' if direction == 'F21'
22 'J68' to 'F20M' if direction == 'J27'
23
24 """
```

<https://github.com/mhvk/scintellometry/blob/master/scintellometry/lofar/digitize.py>

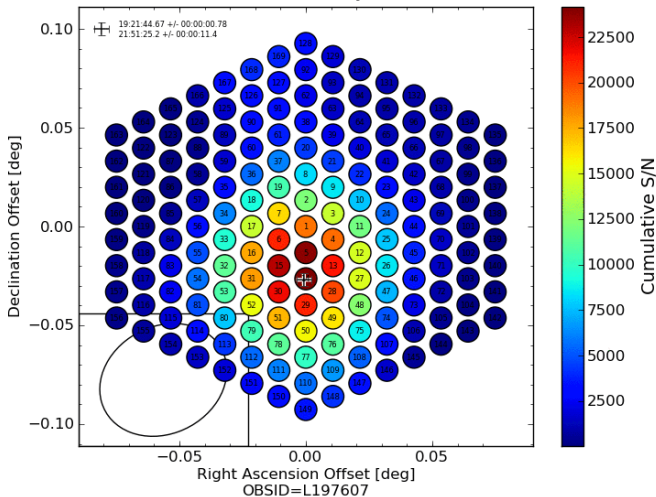
Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]



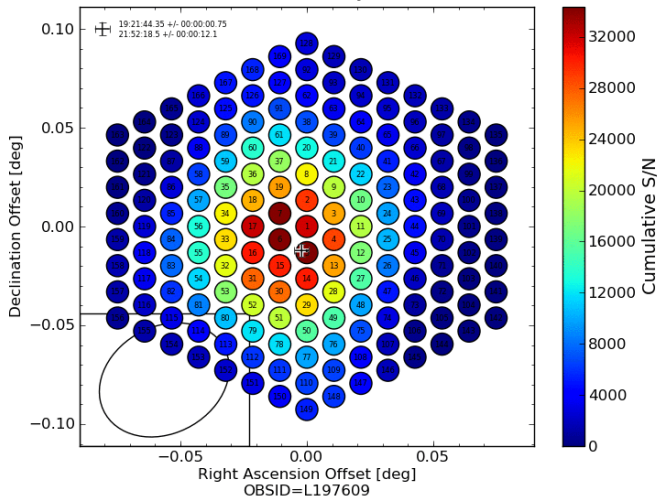
Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]



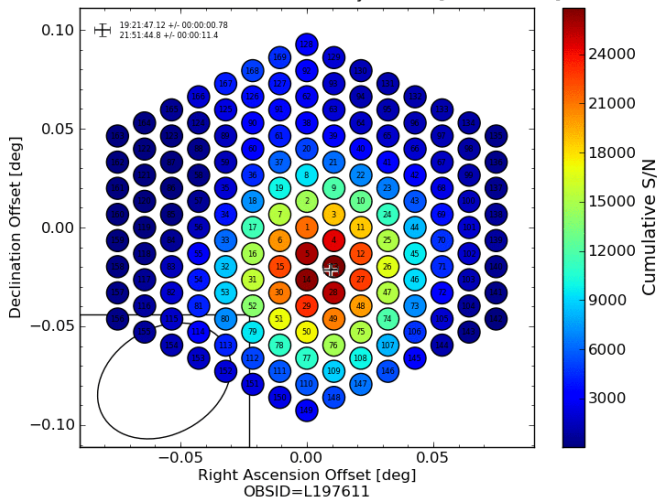
Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]



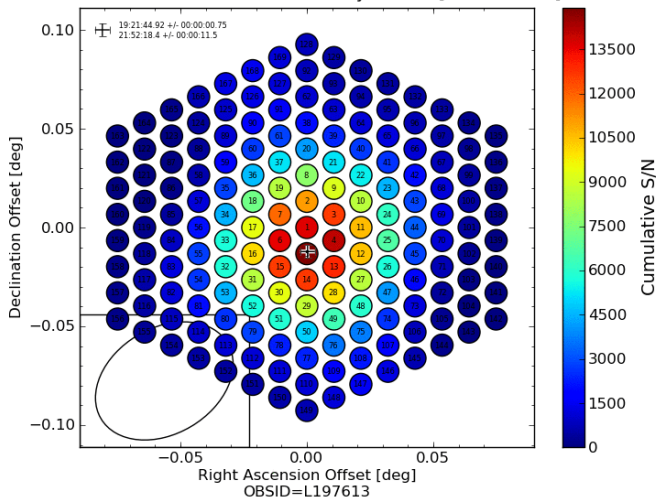
Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]



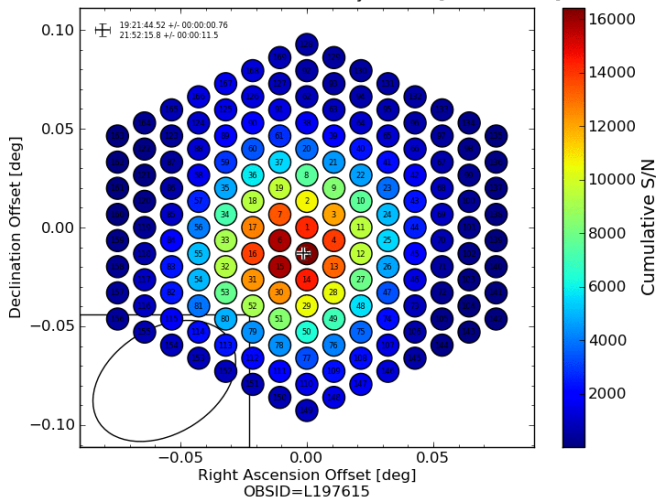
Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]



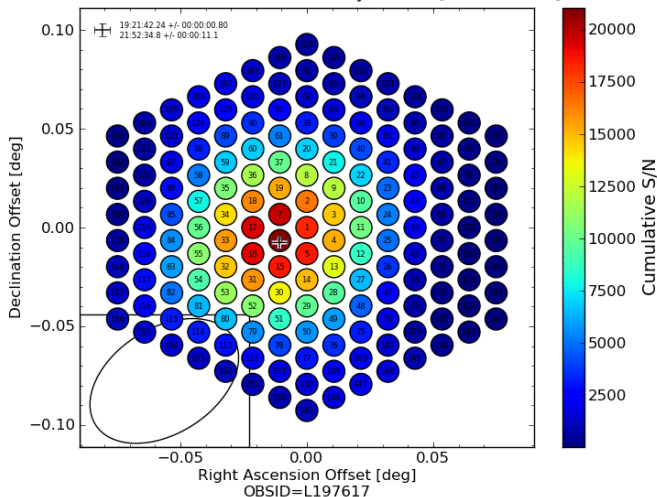
Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]



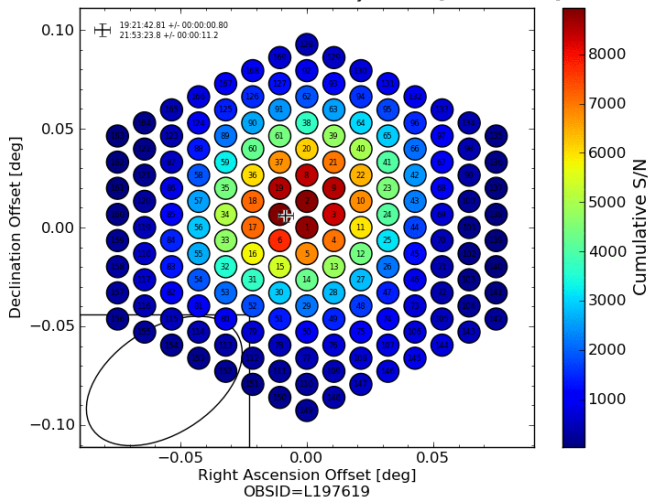
Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]



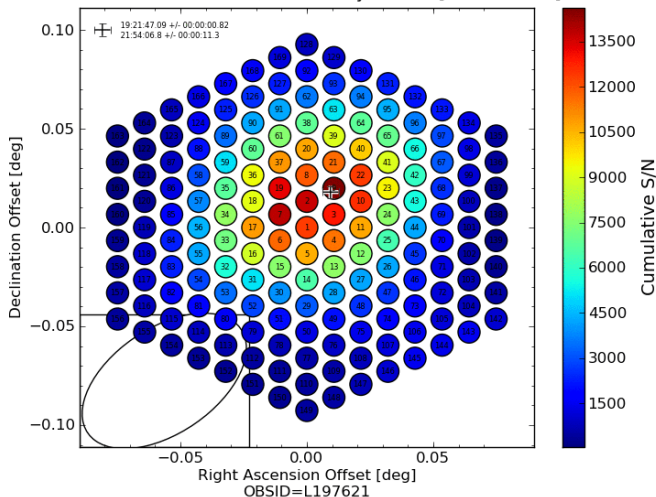
Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]



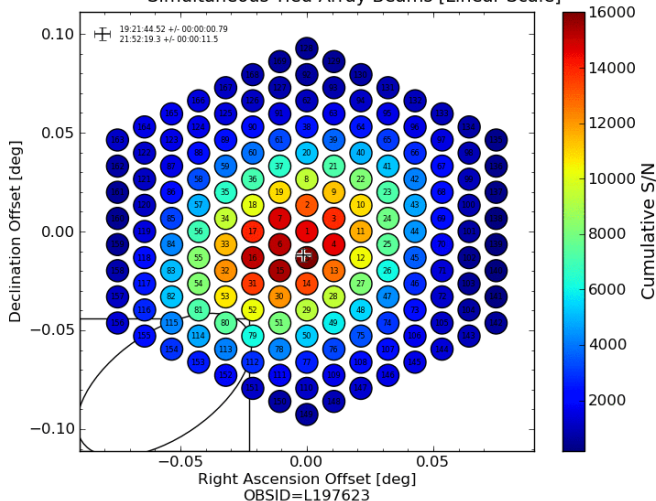
Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]

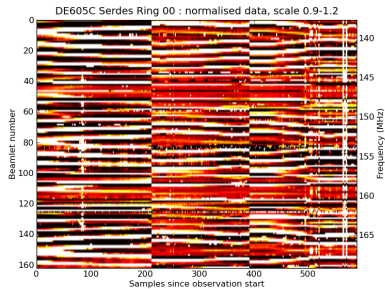
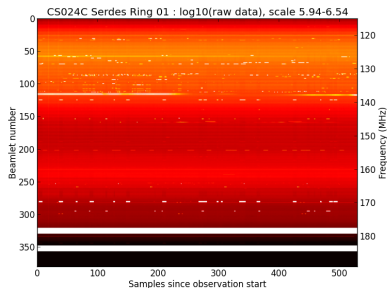


Impact of ionosphere

SAP #0. Cumulative S/N of PSR B1919+21 in 169 (out of 169)
Simultaneous Tied-Array Beams [Linear Scale]



Oscillating tiles



Tutorial 9: Beamformed data inspection

Goal: read and inspect beamformed COBALT output

Requirements: git to download the notebooks, and Python 3, jupyter with numpy, matplotlib and h5py

- **Installing python & jupyter:** Download and install Anaconda 3 from www.anaconda.com
- **Installing h5py:** `pip3 install h5py`
- **Downloading notebooks:**
`git clone https://github.com/cbassa/lofar_bf_tutorials.git`
- **Downloading HDF5 data:**
`ftp://ftp.astron.nl/outgoing/bassa/dataschool/`

